

TLB Translation Setup for MPC603e and MPC755 Microprocessors

by *Amanuel Belay*
Computing Platform Division
Freescale Semiconductor, Inc.
Austin, TX

This application note describes the process of setting up the on-chip table lookaside buffers (TLBs) for translation, without using page tables, for classic PowerPC™ processors with multiple areas of memory mapped one-to-one. The application note applies only to MPC603e and MPC755 (with software table search enabled).

1 Introduction

TLBs are the on-chip “caches” for page table entries for classic PowerPC memory management units (MMUs). On processors that support software table searching and on-chip TLBs, you can set up the TLBs indirectly and use them as extra BATs without the need to set up page tables in memory. The source code works exclusively on the MPC755 and MPC603e (with modifications discussed in the comments). On processors with software table searching enabled, after a TLB miss exception is taken, the interrupt handler can load a TLB entry for the offending address by executing a **tlbld** (or **tlbli**) instruction. For MPC603e and MPC755 processors, the TLB entry information that is loaded is contained in IMISS/DMISS, ICMP/DCMP, and RPA registers. This application note describes how you can load the on-chip TLBs by using these registers as conduits of information with some restrictions (see [Section 5, “Limitations”](#)).

Contents

| | | |
|-----|-----------------------------------|---|
| 1 | Introduction | 1 |
| 2 | Terminology | 2 |
| 3 | Types of Translation | 2 |
| 4 | Configuring On-Chip TLBs | 3 |
| 4.1 | Enabling Software Table Searching | 3 |
| 4.2 | Invalidating the TLB Entries | 3 |
| 4.3 | Configuring the Segment Registers | 4 |
| 4.4 | Loading the TLB Entries | 5 |
| 5 | Limitations | 7 |
| 6 | Document Revision History | 8 |

2 Terminology

The following terms are used in this document:

- BAT Block address translation mechanism. A set of registers that contain the translation information and access privilege for blocks of memory.
- MMU Memory management unit. This on-chip unit manages memory accesses on a processor.
- MSR Machine state register. Contains information on various states of the processor.
- Page 4 Kbytes of contiguous memory starting at a 4-Kbyte boundary.
- PTE Page table entries. These 8-byte entities hold translation information for 4 Kbytes of memory (a page).
- SRR1 Machine status save restore register 1. This register stores information when an exception is taken.
- TLB Translation lookaside buffers. These on-chip storage elements store (cache) recently accessed PTEs.

3 Types of Translation

Processor-generated memory accesses require address translation before they go out to the memory subsystem. Instruction and data access translations are enabled through two bits (IR and DR, respectively) in the machine state register (MSR). When translation is disabled, the processor is said to be in real addressing mode. In this mode, all memory is mapped one-to-one with effective memory/cache attributes (WIMG) settings of 0001 or 0011. When translation is enabled, address translation is performed either through BATs or page tables/TLBs. [Figure 1](#) summarizes the translation types.

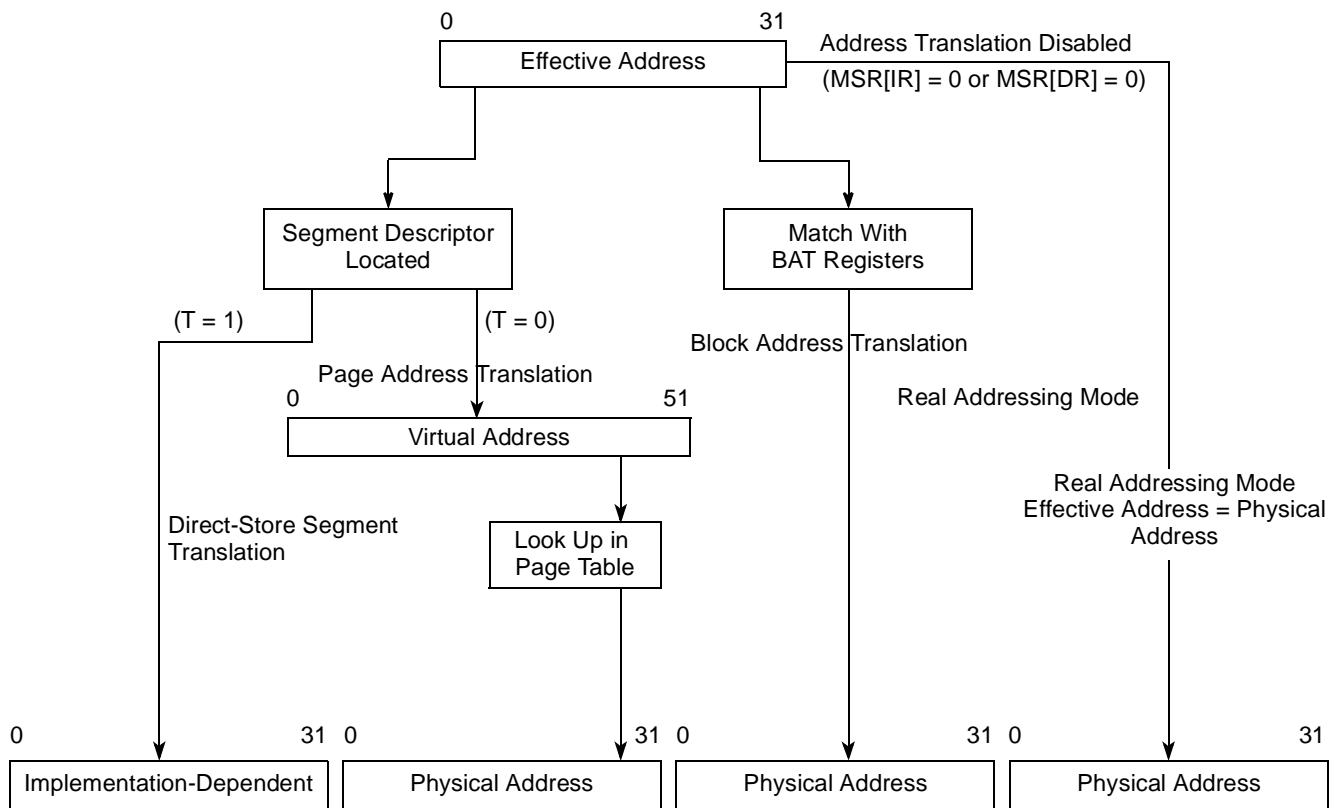


Figure 1. Address Translation Types

For details on the translation types, refer to the *Programming Environments Manual for 32-Bit Implementations of the PowerPC™ Architecture (PEM)*.

4 Configuring On-Chip TLBs

The following sections describe the steps taken to set up the on-chip TLBs when software table searching is used. Note that the MMU should be off (MSR[IR], MSR[DR] cleared) when this setup is run. After setup completes, the MMU is turned back on.

4.1 Enabling Software Table Searching

Software table searching should be enabled (HID2[SWT_EN]=1) in order to perform the steps described in the following subsections.

```
//enable software table searching HID2[SWT_EN]=1
mf spr    r3, 1011
or is     r3, r3, 0x8
mt spr    1011, r3
sync
```

4.2 Invalidating the TLB Entries

Before programming the on-chip TLB entries, you should invalidate all of them first to ensure that only the ones programmed later are valid. The following assembly code invalidates both ways of the on-chip TLBs for a two-way, set-associative TLB array.

```
//invalidate all TLB entries
        li      r5, 64                //change to 32 for MPC603e
        mtctr   r5
invtlb1:
        tlbie   r3
        addi    r3, r3, 0x1000
        bdnz   invtlb1
```

4.3 Configuring the Segment Registers

The segment registers contain the virtual segment IDs (VSIDs) for a process. The upper four bits of the effective address specify which segment register to use. If more than one segment register is used, each must have a unique VSID. To accomplish this, the following code loads the VSIDs with consecutive numbers. In this code, r3 and r4 contain the starting and ending addresses, respectively, of the memory area to be covered by the TLBs.

```

        //set up SRx
        rlwinm    r8, r3, 4, 28, 31        //extract 4 MSBs
        rlwinm    r9, r4, 4, 28, 31        //extract 4 MSBs

srx_set:
        bl      set_srx                    //expects r8=value r9=SR index
        addi    r8, r8, 1
        cmpw    r8, r9
        ble     srx_set
    
```

Where set_srx is defined as:

```

        //set srx registers
        .global set_srx

set_srx:
        cmpwi   r9, 0
        beq     mtsr0
        cmpwi   r9, 1
        beq     mtsr1
        //fill in the same sequence for SR2 up to SR14 here
        cmpwi   r9, 15
        beq     mtsr15

mts0:
        mtsr    0, r8
        blr

mts1:
        mtsr    1, r8
        blr
        //fill in the same sequence for SR2 up to SR14 here

mts15:
        mtsr    15, r8
        blr
    
```

4.4 Loading the TLB Entries

TLB entries have a format similar to PTEs, which are shown in [Figure 2](#). To load the on-chip TLB, the upper and lower portions of a TLB entry are loaded to different registers. Then a **tlbld** (TLB load data) or **tlbli** (TLB load instruction) is executed.

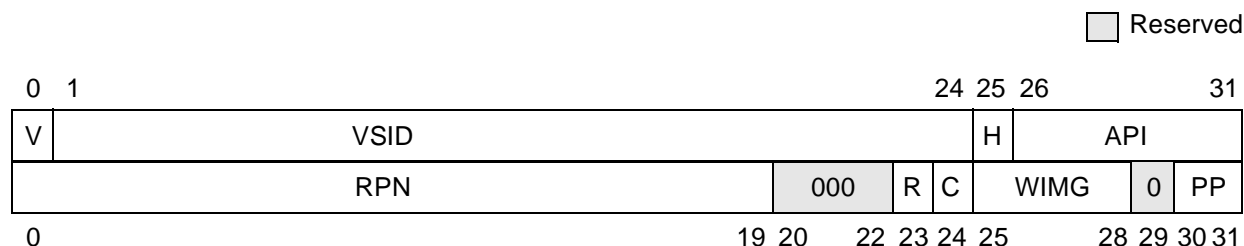


Figure 2. Page Table Entry Format

Each TLB entry is selected by EA[14–19] or EA[15–19] depending on how many of them there are (128 and 64 TLB entries, respectively). All TLBs on the MPC603e and MPC755 devices are two-way, set-associative, and SRR1[WAY] selects the way that is written on a **tlbld** or **tlbli** instruction. Take care not to write to a TLB entry more than once (see caveats in [Section 5, “Limitations”](#)).

4.4.1 Way 0 Selection and Loop Setup

Before we load TLB entries for each of the 4 Kbytes of memory address, we set up the way and loop counter. The loop counter counts up to 64 (for MPC755) or 32 (for MPC603e). In the following code, r3 and r4 contain the starting and ending page addresses, respectively, of the memory area to be covered.

```
//set SRR1[14/WAY] = 0
    li      r5,0
    mtsrr1 r5
    sync

    //initialize counter to 64 (or 32)
    li      r5,64                //change to 32 for MPC603e
    mtctr  r5
    ///LOAD TLB entries

ldtlbs: cmpw   r3,r4
        bgt   exit_ldtlbs
```

4.4.2 The IMISS/DMISS Registers

These registers contain the effective page address of an access that caused a TLB miss. For our purpose we load these registers with the effective address before executing the **tlbld** (or **tlbli**) instruction. In this code, r3 contains the effective address.

```
//set DMISS = EA (for data) or IMISS = EA (for instruction)
    mtspr  976, r3                //change to "mtspr 980, r3" for IMISS
    sync
```

4.4.3 The ICMP/DCMP Registers

These registers contain the upper portion of the PTE/TLB entry upon a TLB miss. For our purpose we load these registers with what we want in the upper portion of the PTE/TLB. In this code, r3 contains the effective address.

```
//set ICMP/DCMP = 0x800000 || 0b00 || EA[4-9]
rlwinm r5,r3,10,26,31
oris r5,r5,0x8000
mtspr 977, r5 //change to "mtspr 981, r5" for ICMP
sync
```

4.4.4 The RPA Register

This register contains the lower portion of the PTE/TLB entry upon TLB miss. For our purpose we load it with what we want in the lower portion of the PTE/TLB. We set up a WIMG of 0000 and R and C are both set to indicate that the processor doesn't need to perform any PTE updates in the page table. This is important because in this setup we do not have page tables in memory.

```
//set RPA = PA[0-19] || 0x182
rlwinm r5,r3,0,0,19
ori r5,r5,0x182
mtspr 982, r5
sync
```

4.4.5 Executing the tlbld/tbli Instruction

After setting up the IMISS/DMISS, ICMP/DCMP and RPA registers, the **tlbld** (or **tbli**) instruction is executed to load the information into the on-chip TLB. The **tlbld/tbli** instruction takes the effective address as a parameter.

```
//load tlb entry indexed by EA[14-19] or EA[15-19]
tlbld r3 //or tbli for instruction TLB
sync //just to be safe
```

4.4.6 Way 1 Selection and Loop completion

After loading Way 0 entries, we switch to Way 1. The address range of the memory should not exceed 64 pages (or 32 pages) for each way (see restrictions in [Section 5, "Limitations"](#)).

```
//increment EA/PA by 0x1000
addi r3, r3, 0x1000
bdnz ldtlbs //continue if counter > 0
//if we filled all 64 entries of one way, toggle to the second way
//reset counter
li r5,64 //change to "li r5, 32" for MPC603e
mtctr r5
```

```
//toggle SRR1[14/WAY]
mfsrr1 r5
xoris r5,r5,0x2
mtsrr1 r5
sync
//continue
b ldtlbs
```

5 Limitations

Using TLBs as described in this document has some restrictions. First, this technique applies only to processors that implement on-chip TLBs and that can also use software table walk. The source code works only on MPC755 and MPC603e (with some modifications as indicated in the comments). Second, we do not get to choose which TLB to use. The TLB entry to be used is selected by EA[14–19] or EA[15–19] of the effective address. Therefore, there are limitations on the size of the memory area that can be covered. The memory area to be covered should be within two memory area windows, as follows:

- Each window is contiguous
- Each window is no more than 64 pages (for MPC755) or 32 pages (for MPC603e)

For example 0, 0x40000 and 0x80000 lie over three different 64-page windows of memory. All three windows cannot be covered by the TLB entries (128 of them) at the same time. The two ways of TLB entry 0 can handle two of the addresses, but the third address cannot have a TLB entry without overwriting one of the ways.

0, 0x3F000, and 0x40000 can be covered by the TLBs at the same time on the MPC755 (which has 128 TLB entries) but not the MPC603e (which has 64 TLB entries). 0 and 0x3F000 lie within 64 pages of each other (using TLB entry 0 and TLB entry 63, respectively, and 0x40000 is covered by Way 1 of TLB entry 0).

6 Document Revision History

Table 1 provides a revision history for this application note.

Table 1. Revision History

| Revision | Date | Substantive Changes |
|----------|---------|---------------------------------|
| 0 | 10/2004 | Initial public release |
| 1 | 5/2006 | Corrected error in Section 4.2. |

THIS PAGE INTENTIONALLY LEFT BLANK

How to Reach Us:

Home Page:

www.freescale.com

email:

support@freescale.com

USA/Europe or Locations Not Listed:

Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
(800) 521-6274
480-768-2130
support@freescale.com

Europe, Middle East, and Africa:

Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:

Freescale Semiconductor Japan Ltd.
Technical Information Center
3-20-1, Minami-Azabu, Minato-ku
Tokyo 106-0047 Japan
0120 191014
+81 3 3440 3569
support.japan@freescale.com

Asia/Pacific:

Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate,
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:

Freescale Semiconductor
Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
(800) 441-2447
303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@
hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters which may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. The described product is a PowerPC microprocessor. The PowerPC name is a trademark of IBM Corp. and used under license. All other product or service names are the property of their respective owners.

© Freescale Semiconductor, Inc. 2004, 2006.

AN2795
Rev. 1
5/2006